Impact of Security Factors in Software Project Risk Assessment using Neural Networks

Subhashis Pradhan

Roll. 710cs2038

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela – 769 008, India

May 2015

Impact of Security Factors in Software Project Risk Assessment using Neural Networks

Dissertation submitted

to the department of

Computer Science and Engineering

(Specialization in Information Security)

of

National Institute of Technology Rourkela

in partial fulfillment of the requirements

for the degree of

Masters of Technology

by

Subhashis Pradhan

(Roll. 710cs2038)

under the supervision of

Prof. S. K. Rath

Department of Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela – 769 008, India

May 2015

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India.

Date: May 29, 2015

Certificate

This is to certify that the work in the thesis entitled *"Impact of Security Factors in Software Project Risk Assessment using Neural Networks"* by *Subhashis Pradhan*, bearing the roll number 710cs2038, is a record of an original research work carried out by him under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of *Masters of Technology* in *Computer Science and Engineering Department (Specialization in Information Security)*. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

Prof. Santanu Kumar Rath
Department of CSE
NIT Rourkela, India

Acknowledgment

First and foremost, I would like to express my deep sense of respect and gratitude towards my supervisor Prof. Santanu Kumar Rath, who has been the guiding force behind this work. I would like to thank him for introducing me to the field of Risk Management and Neural Networks and giving me the opportunity to work under him. His unparalleled knowledge in this topic and ability to bring out the best of analytical and practical skills in people has been invaluable in tough periods. Without his invaluable advice and assistance it would not have been possible for me to complete this thesis. I am greatly indebted to him for his constant encouragement and invaluable advice in every aspect of my academic life. I consider it my good fortune to have got an opportunity to work with such a wonderful person.

I thank Mr. Shashank Mouli Satapathy, for his constant support in my thesis work. He have been great a source of inspiration to me and I thank him from the bottom of my heart.

I would like to thank all the faculty members and secretarial staff of the CSE Department for their sympathetic cooperation.

Subhashis Pradhan

Abbreviations

COCOMO : Constructive Cost model

VL : Very Low

L : Low

H : High

VH : Very High

XH : Extremely High

ExCOM : Expert COCOMO

RBFNN : Radial Basis Function Neural Network

FCM : Fuzzy C- Means

MMRE : Mean magnitude of relative error

MSE : Mean squared error

GA : Genetic Algorithm

SDLC : Software development life cycle

EAL : Evaluation Assurance Level

ISO/EIC : International Organization for Standardization/International Electrotechnical Commission

NASA : National Aeronautics and Space Administration

Abstract

"software risk" is the measurement of the probability of an unwanted output that could affect the software product's development process. It always includes the chance of being uncertain and a potential for loss. This paper extends the concepts of Constructive Cost Model (COCOMO) model into fuzzy Expert COCOMO by introducing security factors as additional parameters for the assessment of risk of a software project. This approach is validated with the NASA60 project data and proved that Genetic Algorithm provided efficient risk values with different levels of security parameters. However, in the earlier methods, there was a limitation in effectively dealing with linguistic forms of imprecise and uncertain inputs. This resulted in increase in the cost of designing the mechanisms for security purposes, that formed a major part in the overall cost in the development process of the software product. The risk value of a software project could well be reduced by taking security factors into consideration. The neural network techniques used for validating the risk values are Kohonen neural network, Radial Basis neural (RBF) network, Learning Vector Quantization, Genetic Algorithm(GA). A comparison study has been provided for all the neural network models implemented in order to examine their performances.

Keywords: fuzzy, learning, classification, epochs, risk, risk-control, and risk-assessment.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Risk Assessment

Prioritizing the risks so as to focus the attention and resources on the more risky items is the goal of risk assessment. The beginning process in risk assessment, risk identification, in which all the identification of the different types of risks for a software project is done [2]. Exercising in envision about what can go wrong, the risks are made project dependent and identified. Work products, processes, plan reviews, meetings and brainstorming, surveys and checklists of all the possible risks are included in the methods that aid risk identification. Personnel shortfalls are the top ranked risk items. Rather than having a project with people having specific skills, it involves having fewer people than necessary [3]. Matching the needs of the project with all the skills available and getting the top talent possible are some of the ways to manage this risk. In order to reduce the risk along with acquiring key personnel for the critical areas of the project adequate training is required, which will reduce this risk. Unrealistic budgets and improbable schedule, that happens so frequently because of business and other reasons is the second item. Imposing a schedule for a software project that has no characteristics of the project on the high-management is very common. It is also considered unrealistic. As a result of

inexperience and optimism, there is a chance that underestimation may occur [4]. Requirements are related to the next few items. If the analysis of requirements is not done properly or if the development process is started too early, projects generally run the risk of developing the wrong software. Development of user interface may be considered similarly. As clients are hesitant to utilize it, extensive rework or the client interface or software perks may not be acquired later.

Addition of features into the software product that are only useful by a margin is known as "gold plating". As gold plating causes consuming resources and time with little returns, unnecessary risk is added to the project. Enumerating the unforeseen events that might occur, identification of the unwanted events that might occur during the development during the project is done in risk identification. Neither the impact on the project nor the specification of probabilities of these risks can materialize, if the risks indeed materialize. Hence risk analysis and prioritization are the next tasks.

Both the scenarios, loss due to materialization of risk and the chance of occurrence of estimated risk is taken into account in risk analysis. Though the concept of structured approaches also exists, this is generally done with detailed discussion, using the understanding and the user experience of the situation.

1.2 Risk Control

Identifying few risk items in the top of the list and focusing on them is the ultimate motive of risk management [5]. Once the risks have been identified and prioritized by the project manager, identification of the top priority risks can be done easily. The main question would then be what to do with them. The basic and ultimate aim of management of risk is to know the value of risk and if one can plan a flow chart so that there are minimal consequences. Reducing the loss due to risk materialization or reducing the chance of materialization of risk should be the strategy for performing actions for most risks. These are known as "risk mitigation steps". In order to

decide whether to take a step or not a list should be maintained that would contain all the mitigation steps which may be useful for the various types of risks. During the time risk analysis is performed, risk perceptions should generally have a basis of prioritizing the risks and consequent planning. As these risks are events based on probability, that depend on factors externally, the threats due to risks may change with time as the factors undergo change. Then the perception of risk changes with time clearly. Besides, the factors that may affect the perception of risk may be undertaken for the risk mitigation steps [6].

Periodic reevaluations, regular monitoring should be the dynamism for the treatment of risk evaluation. Thus, a software project must be revisited periodically to evaluate the risk perception and modifying the risk mitigation plans in addition to monitoring the planned risk progress. Looking into the various risks' status and controlling their activities is known as risk monitoring. Altering the plans as needed in the process, and analyzing the risks periodically at each checkpoint is one of the many approaches for risk monitoring.

1.3 Applications of Security in Software Projects

There has been a tremendous change the way organizations tend to align with the use of Information Technology in their business. The evolving e-business trend has taken enterprise to its competitive edge. Business Applications have been evolving at a rapid pace to serve business needs. Applications are those that execute on distributed environment or any real time applications (mostly Web based or any Client Server applications with multiple functionalities to server the requirements. Due to this rising trend in use of these applications, management tends to procure or develop applications at a faster rate. Application architects and developers are more aligned to the functionality of the applications developed. Management is satisfied since most of the applications sever and support their business needs. Hence the application development process is also aligned to the functionality and as always,

security-need on these applications has come down in the priority list.

Following the standard SDLC phases and aligning to the engineering principles of software project, most application developers, never consider or follow a disciplined process in order to address the security factors in any of the development phases. In any application does the authentication and authorization mechanisms follow the secure mechanisms (such as login and password). Does the developed application help in addressing the security in entirety. Functionality is listed in the same priority on the basis of the fact that security attacks at the application layer have developed a need for security parameters. This paper clarifies about how Security can be combined or recognized in the Software Engineering principles (SDLC phases) and how an Organization can influence after considering Security as a powerful process inside of the current development framework.

1.4 Problem Definition

For the software project development to be successful, the initial Project Planning phase is the most important phase, as it involves numerous activities that help in determining a project's cost, risk, scope, scheduling and the available resources. Due to the uncertain data , complexity of the underlying process, and the intangible nature of the product, a software project can sometimes be risky. Though it is difficult and expensive to implement, management of risk is mandatory in the process of development of software products. Of all the various activities, the planning phase of software project can be clustered into numerous activities, some of which are estimation of effort and management of risk. Requirement of effort in a software project, is calculated in software effort estimation [7], based on the several factors related to cost. Management of Risk, on the other hand, helps in identifying, addressing and eliminating all or most of the software project risks before any unwanted events occur. There is a difficulty in implementation and practice in risk management as compared to effort estimation [8]. Another important

topic that is connected with software development, is the system security, which includes development of data systems with the usage of IT. The security systems have been constantly evolving since the past few years. Sensitive data, information and knowledge are often stored and processed by the information systems, and they must have a proper security resistance from the unwanted monitoring and attacks. The development of data systems should include design and subsequent implementation of appropriate security mechanisms. The total cost of development of software projects with high security demands include a major cost of implementation of security mechanisms. The phase of project development cost estimation should include the cost for data systems security for better cost efficiency.

The international standard ISO/IEC 15408 defines the informational technologies and are evaluated by Common Criteria. The criteria results in a balanced perspective due to the considered product′s correctness, and is comprised of special purpose sets. The criteria also specifies Evaluation assurance level.

ISO/EIC [9] defines security levels as follows:

EAL1: functionally tested : Being the lowest form of assurance level, it provides a trust to a certain amount and it does not use the complex analysis of security risks. The definition of interface, specifications of functions and security process documentation.

EAL2: Structurally tested : Independent testing extends the requirements of EAL2, as compared to EAL1. In order to deal with the product security attacks, there is a need to extend the development of product due to the description of informal architecture at the same time. At times when the entire set of data about the development phase is not made available, low to medium level verified security is provided independently by the considered level.

EAL3: Methodically Tested and Checked : without doing any increase in the values of effort, the proven ways or approaches allow the maximal assurances of the development process. Ensuring a request to the administration configuration, monitoring the environment development, and testing a wider range of security

functions and mechanisms is availed as compared to EAL2. For the independent verification of security in a medium level, this level is recommended.

EAL4: Methodically Designed, Tested and Reviewed : It is a highly reliable level that is basically based on the all the development methods related to quality. However any extensive knowledge regarding specific knowledge involvement, skills or resources is not required. And with low attack potential against attackers, a vulnerability analysis must be proven independently.

EAL5: Semi-Formally Designed and Tested : For a medium extent in security engineering, usage of special methods and techniques are required by this level during development. For the overall design and concealed channel analysis a semi-formal representation is required which has to be completed the assurances of the formal security model. Using the medium performance of the sources, the resulting products can resist any type of attacks.

EAL6: Semi-Formally Verified Design and Tested : It is a design that is solely based on the modular and layered design, this level helps in developing a controlled environment that is strictly managed and is highly resistant to attacks. Systematic analysis must be performed by the concealed channels.

EAL7: Formally Verified Design and Tested : The level is based on the highly demanding design, this level is formally designed. This results in the decrease of design complexity. Correspondence formal, semi-formal presentation, detailed semi-formal design, presentation of a formal specifications of functions and global design, a formal model of security policy and full formalization of the level is required.

1.5 COCOMO Model for Software Risk Assessment

Constructive Cost Model II (COCOMO II) [10] helps in cost estimation,effort estimation and scheduling of activites in software development activity. COCOMO II is the most recent real expansion to the first (COCOMO 81) [11] model

distributed in 1981. COCOMO-II has 15 effort multipliers namely,

1) acap : capability of analysts

2) pcap : capability of programmers

3) aexp : experience of application

4) modp : modern programming practices

5) tool : software tools usage

6) lexp : experience in language

7) sced : constraint for schedule

8) stor : constraint for main memory

9) data : size of data base

10) time : cpu time constraint

11) turn : turnaround time

12) virt : volatility of machine

13) cplx : complexity of process

14) Rely : reliability of required software

15) Vexp : experience in virtual machine

Prioritizing of project risks, categorizing, and identifying of project planning are aided by Expert COCOMO, which is an improvement of COCOMO-II. Performing an early stage assessment of risk from the earlier activities of effort estimation by using an existing knowledge is the main advantage of Expert COCOMO.

A 16th multiplier SECU-software security is added in the model to verify the risk of the software. The individual classes form a basis of determination of the influential degree of security factor for the product [1]. The degree of influence is shown in table 1.1

The security attribute varies from very low to extremely high. The mapping of security attribute from fuzzy terms to numerical terms is as shown in table 1.2.

Table 1.1: Mapping of Security Attributes [1]

Influential Degree	**Assurance Level**
0	**EAL-1**
1	**EAL-2,EAL-3**
2	**EAL-4**
3	**EAL-5**
4	**EAL-6**
5	**EAL-7**

Table 1.2: Risk level Assignment Matrix [1]

Attribute	**VL**	**L**	**N**	**H**	**V**	**XH**
Value	**0.90**	**0.94**	**1.00**	**1.19**	**1.36**	**1.88**

1.6 Evaluation Criteria

All the above neural network techniques were analyzed and compared according to the following evaluation criteria [12]:

1. **Epochs**: An epoch is a measure of the no. of times the majority of the training vectors are utilized once to update the value of the weights.

2. **MSE (Mean Squared Error)**: If is a vector of predictions, and is the vector

of the true values, then the (estimated) MSE of the predictor is:

$$MSE = \frac{\sum_{i=1}^{T} (pred_i - act_i)^2}{T} \tag{1.1}$$

where

$pred_i$ = Actual risk value.

act_i = Predicted risk value.

T = Total no. of data.

3. **Correlation Coefficient** : It is used to find how strong a relationship is between attributes. In this study Pearsons Correlation Coefficient formula has been applied which is shown below [13].

$$r = \frac{T(\sum AB) - (\sum A)(\sum B)}{\sqrt{(T\sum A^2 - (\sum B)^2)(T\sum B^2 - (\sum B)^2)}} \tag{1.2}$$

where

r is a Correlation Coefficient. A and B are attributes.

4. **MMRE (Mean Magnitude of Relative Error)**: MMRE is defined as follows

$$MMRE = \sum_{1}^{N} \frac{|x_i - y_i|}{x_i} \tag{1.3}$$

where

x_i = actual risk value,

y_i = predicted risk value

5. **Pred(m):**Another broadly utilized prediction value is pred(m), which is essentially the rate of evaluations that are inside m/100 of the genuine worth.

Here m is situated to 0.25, 0.5 and 0.75.

$$Pred(m) = \frac{\sum_1^N S}{V} \tag{1.4}$$

where,

$$S = 1, if Pr \pm \frac{m}{100} \tag{1.5}$$

$$S = 0, if PR > \frac{m}{100} or PR < \frac{m}{100} \tag{1.6}$$

and V is the total number of values.

1.7 Thesis Outline

Organization of the thesis is outlined below:

In chapter-2, risk assessment, risk control and the effects of risk in a software project is described.

In chapter-3, all the neural network techniques used for the assessment of impact of security factors for the assessment of risk in a software product is described.

In chapter-4, comparison of all the techniques is done.

Finally Conclusion and future work are described in chapter-5.

Chapter 2

Literature Review

2.1 Review of work on Risk Assessment

Barry Boehm [14] described emerging discipline of software risk management. He has identified various risk assessment models with support of implementation details to validate a model.

A. V. Deursen et al. [15] assessed the project risk based on facts available for project. The facts include software project size, development effort. He has also taken account of people working on the project and documentation available for project. He described how this facts are interpreted properly to assess the project risk.

Daya Gupta et al. [16] worked on the project risk due to failure of project or over budget. They had proposed a model for estimation and assessment of risk. This model is efficiently accurate in predicting risks involved in software project. The Mission Critical Requirements Stability Risk Metrics are used in this paper for estimating risk. This model assesses risk for every phase of development cycle of the software.

Mark Keil et al. [17] analyzed the risk of a software by mapping the different types of risks identified in a 2x2 grid.

2.2 Review of work on Neural network based Risk Assessment

Green et al. [18] discussed at lengths about the applications of neural network techniques in estimating the risk of management fraud.

Lyu et al. in their book [19] explained how the neural network models can be applied to the software project modules.

Zan Huang et al. [20] in their article explained the use of neural networks in stock market predictions. The models use the stock price of the companies from the market and estimate a value for the coming days or years.

Thomas G Calderon et al. [21] in their paper focused on the utilization of neural systems as a foundation for a new business architecture and provides understanding into forthcoming research opportunities.

Briand et al. [8] in their paper compared numerous neural network techniques that could be used for both effort estimation and risk estimation. Constructive Model (COCOMO) model was used for the training and assessment of risk and effort.

Boehm et al. [22] in his paper summarized various software cost estimation techniques, some of which included regression-based models, parametric models, composite Bayesian techniques etc.

M. V. Goyal et al. [23] proposed a neuro-fuzzy technique based on cost drivers for assessing software project risks. They found their proposed implementation result exhibits better performance than traditional fuzzy-ExCOM technique.

2.3 Review of work on security based Techniques

Haag et al. [24] stated the necessary steps that needed to followed for developing a secure software product.

Christian Sven Collberg et al. [25] gave obfuscation methods to upgrade software security. In one exemplification, a technique for obfuscation procedures for

upgrading software security incorporates selecting a subset of code (e.g., assembled source code of an application) to jumble, and obfuscating the selected subset of the code.

Andreas L et al. [26] in their paper stated that the security requirements should be made elicit so as to reduce the misuse of software products.

Chapter 3

Methodologies Used

3.1 Kohonen Neural Network

Kohonen's systems [27] are one of the fundamental self-organizing neural systems. The capacity to self-organize gives new conceivable outcomes - adjustment to earlier obscure data inputs. It is by all accounts the most normal method for learing, which is utilized as a part of our brains, where there was no characterization of patterns. Those examples come to shape amid the learning procedure, which is consolidated with ordinary work. Kohonen's systems are an equivalent word of entire group of networks which make utilization of self-organization, competitive type learning system. One sets up weights on network's inputs and after that pick a winning neuron, the particular case that compares with input vector in the most ideal way. Exact plan of competition and later changes of hidden layers may have different structures. There are numerous sub-layers in view of competition, which contrast themselves by exact self-organizing output data vector calculation. Working of self-arranging neural system is separated into three stages:

1. Construction

2. Learning

3. Identification

Framework, which should realize the working of self-organizing system, ought to comprise of couple of fundamental components. These signals ought to describe a few qualities of effects which occur in the surrounding. On account of that description the network has the capacity to gather those effects.

Fig.3.1 shows the architecture of Kohonen Neural Network.

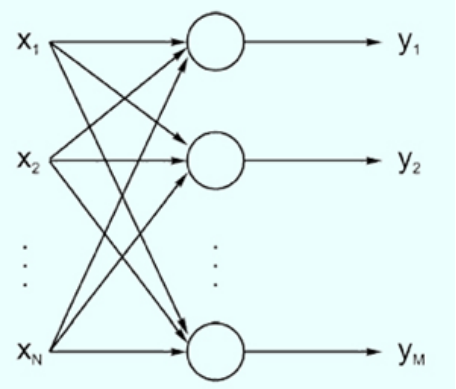

Figure 3.1: Kohonen neural Network

Finally, completely vital for self-organizing procedure is that the network has the capacity to adjust weights of winning neuron and its neighbors, as per response quality. Net topology can be characterized in an extremely basic manner by deciding the neighbors of each neuron. One should call the unit whose reaction on weights is maximal the winner of this learning procedure. At that point we can assume that the network is all together, if topologic relations between data signals and their weights are identical.

The Kohonen neural framework contains 2 layer of neurons, an input layer and an output layer. Kohonen neural system does not contain any hidden layer. As a matter of first importance, one inspect the input layer and output layer to a Kohonen neural framework.

The name of the entire class of systems originated from the assignment of calculation called self-organizing Kohonen's maps. They had been portrayed in

the publication "Self Organizing Map". Kohonen proposed two sorts of proximity : "rectangular" and "gauss". The principal is:

"lambda" is the range of closeness, it diminishes in time. Utilization of Kohonen's strategy gives us preferable results over "Winner Takes All" method. Association of the work is better (neurons association speaks to the dispersion of input data in a superior manner) and the merging of the algorithm is higher. Due to that the time of single iteration is a couple times longer, where weights of numerous neurons , not just winners', must be altered.

3.2 Radial Basis Function Neural Network (RBFN)

RBFN [28] compare to a specific class of function approximation which can be prepared, utilizing an arrangement of a set of samples. RBFNs have been getting a developing measure of consideration since their proposition, and now a lot of hypothetical and observational results are accessible [7].

The rough guess procedure utilized as a part of RBFN comprises of approximating an unknown function with a linear comination of nonlinear functions, called "radial functions". Utilization of Kohonen's system gives us preferable results over "Winner Takes All" technique. Association of the network is better (neurons association represents the distribution of input data in a superior manner) and the convergence of the algorithm is higher. As a result of that the single iteration is a couple times longer - weights of numerous neurons, not just winners', must be altered.

Fig .3.2 shows the architecture of a typical RBFN.

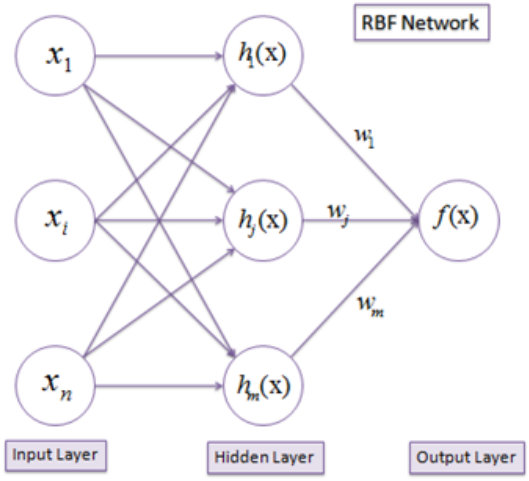

Figure 3.2: Radial Basis Neural Network

3.3 K-Means RBFN (Radial Basis Function Network)

Proposed around 1975 and 1977 by J. A. Hartigan and M. A. Wong [29], K-Means clustering is one of the more established predictive modeling techniques.It is a moderately quick modeling technique, yet it is likewise among the slightest exact models that DTREG [30] offers.

Typically, both RBF systems and PNN systems provide more accurate results as compared to the K-Means clustering models. PNN systems have the highest accuracy of all methods, yet they turn out to be illogically slow in case there are more than around tens of thousands of rows in the training data file. K-Means clustering has high computation speed as compared to RBF or PNN systems, and it can deal with huge training files [31].

K-Means clustering can be used only for classification (with an absolute target variable), and not for regression. The target variable may have two or more classes. To comprehend K-Means clustering, one should consider an arrangement including two objective classifications and two indicator variables.

Creating a K-Means clustering model has two major problems:

1. Determination of the number of optimal clusters to be created.

2. Determination of cluster center.

With the number of clusters specified, the next part of the issue is figuring out where the center of each cluster should be placed. Frequently, centers are scattered and don't fall into effectively recognizable clusterings. Cluster center determination is done in two stages:

1. Starting positions for the clusters are at first determined. This is done in two steps:

   (a) The first center is assigned to a random point.

   (b) Untill the optimation is achieved, the center positions should be adjusted.

2. The most distant existing center is determined from an existing center and the assigned to the next center to it.The process is repeated until the specified number of cluster centers have been found.

3.4 Random Centers RBFN

A random centers Radial Basis Function Network (RBFN) [32] is a specific type of neural system. A RBFN performs classification by measuring the inputs's similarity to patterns from the training set. Each RBFN neuron stores a "prototype", which is only one of the patterns from the training set. When we need to classify another input, every neuron calculates the Euclidean distance between the input and its framework. Generally speaking, if the inpiut more nearly takes after the class A models than the class B models, it is delegated class A.

For irregular centers, the center is taken as random numbers in the range of the input values. The number of centers are predefined by the user. The values of the centers are changed at whatever point there is an adjustment in the gradient of the input vector.

3.5 Fuzzy C-Means RBFN

Fuzzy c-means (FCM) [33] is one of the techniquse for clustering that permits one bit of information to fit in with two or more groups. This system (developed by Dunn [34] in 1973) is frequently utilized as a part of pattern recognition. The objective function given below is based on the minimization of Fuzzy C-Means RBFN:

$$J_m = \sum_{i=1}^{M} \sum_{j=1}^{N} mem_{ij}^{m} (data_i - center_j)^2 \tag{3.1}$$

where m is any real number not less than 1, mem_j is the value of cluster j, s is the membership degree of $data_i$. $data_i$ is the i^{th} measured data whose dimension is d , $center_j$ is the d-dimensional cluster center.

3.6 Learning Vector Quantization

A LVQ system [35] consists of 2 layers, which are, competitive layer and linear layer. In any case, one needs experience a training pattern altogether get the starting layer to convey the right subclass yield for each vector of the training set. In the first and foremost place, one ought to consider how to make the first framework.

3.7 Genetic Algorithms

Genetic Algorithms (GAs) are versatile heuristic search algorithms taking into account the developmental ideas of common choice and genetics. In that capacity they represent an intelligent exploitation of an arbitrary inquiry used to optimization issues. Although randomized, GAs are in no way, shape or form irregular, rather they exploit authentic data to direct the inquiry into the region of better performance inside of the search space. The essential procedures of the GAs are intended to recreate forms in characteristic frameworks fundamental for evolution, uncommonly those take after the standards first set around Charles Darwin of "survival of the fittest.".

GAs reproduce the survival of the fittest among individuals over continuous generations for taking care of a problem. Every generation comprises of a populace of character strings that are closely resembling the chromosome that we find in our DNA. Every individual speaks to a point in a search space and a possible solution. The individual in the population are then made to experience a procedure of advancement.

GAs are in view of a relationship with the genetic structure and conduct of chromosomes inside of a population of people utilizing the accompanying establishments:

1. Individuals in a population go after resources and mates.
2. Those people best in every "competition" will create more posterity than those people that perform inadequately.
3. Qualities from 'good' individuals propagate all through the population so that two great folks will now and again create posterity that are superior to either parent.
4. Subsequently each progressive era will turn out to be more suited to their surroundings.

A population of individuals is kept up inside of search space for a GA, every speaking to a conceivable answer for a given issue. Every individual is coded as a limited length vector of parts, or variables, as far as some random floating point number set, typically the double numbers in order 0,1. To proceed with the hereditary similarity these people are compared to chromosomes and the variables are undifferentiated from qualities. Accordingly a chromosome (arrangement) is made out of a few qualities (variables). A wellness score is allocated to every arrangement speaking to the capacities of a person to 'compete'. The person with the ideal (or for the most part close ideal) wellness score is looked for. The GA plans to utilize particular 'breeding' of the answers for produce 'offspring' better than the folks by joining data from the chromosomes.

3.7.1 Results

The values of all the evaluation criteria of all the neural network models is demonstrated underneath in table 3.1 :

Table 3.1: Experimental Values without considering Security Attribute

	MMRE	MSE	PRED(0.25)	PRED(0.5)	PRED(0.75)	Correlation	Epochs
Kohonen	**0.12**	**0.26**	**0.72**	**0.79**	**0.96**	**0.65**	**226**
K-means	**0.06**	**0.19**	**0.53**	**0.66**	**0.81**	**0.82**	**132**
Random Centers	**0.06**	**0.11**	**0.491**	**0.54**	**0.73**	**0.91**	**91**
Fuzzy C-means	**0.07**	**0.036**	**0.61**	**0.68**	**0.772**	**0.88**	**372**
Learning Vector Quantization	**0.195**	**0.265**	**0.615**	**0.662**	**0.7375**	**0.679**	**105**
Genetic Algorithm	**0.05**	**0.012**	**0.72**	**0.85**	**0.88**	**0.963**	**59**

3.7.2 Conclusion

Table 3.1 shows that applying neural network techniques reduces the risk values in the dataset. This proves that applying neural network provides better result than Expert-COCOMO.

3.8 Applications of neural networks for fuzzy-Excom based risk Assessment with Security Attributes

3.8.1 Steps of Implementation

The Following section explains the steps in process:

1. **Preparation of Data:** COCOMO model consists of 15 cost drivers that are used for effort estimation. An additional cost driver SECU (security factor) is added along with the 15 drivers.

2. **Normalization of datas:** The values of cost factors of project in NASA60 dataset is in linguistic terms. To make this values to feed as input for our model taheir is need to convert it into numerical form. MIN MAX normalization formula is used to do so.

$$norm' = \frac{norm - AMin}{AMax - AMin}(new\_Amax - new\_Amin) + new\_Amin \quad (3.2)$$

3. **Assessment of Risk using Expert COCOMO:** Comprising of a few number of risks, that are identified with COCOMO cost factors, is the risk hierarchy in Expert COCOMO. A few number of cost drivers combine to form the risk values in Expert COCOMO. The risk level matrix helps in mapping the cost drivers in pairs to determine the risk rules.

4. **Risk assessment using neural network techniques:** The neural networks implemented first map all the linguistic values into numerical values. All the 105 values are then given as input to the neural network models for classification and analysis. The training and testing data set consists of 70 and 35 data points, respectively.

5. **Defuzzification:** The process of defuzzification performs the mapping of risk values from a low level to a high level.

6. **Performance Evaluation:** A comparison has been made between the neural network models Kohonen,k-means RBFN,Random Centers RBFN and Fuzzy C-Means RBFN in this paper. The performance of the models is evaluated using final MSE, MMRE, prediction accuracy (PRED) and correlation.

The security attribute varies from very low to extremely high. The mapping of security attribute from fuzzy terms to numerical terms is as shown in table 3.2.

Addition of Security Attribute: The Security attribute is added as the 16th attribute in the dataset. The corresponding numeric values for each of the fuzzy value is mapped using table 3.2.

Table 3.2: Security attribute values [1]

Attribute	VL	L	N	H	VH	XH
Security	0.90	0.94	1.00	1.19	1.36	1.88

Table 3.3: Experimental Values with Security Attribute

	MMRE	MSE	PRED(0.25)	PRED(0.5)	PRED(0.75)	Correlation	Epochs
Kohonen	0.2	0.3	0.6	0.74	0.91	0.73	252
K-means	0.08	0.28	0.492	0.61	0.75	0.89	79
Random Centers	0.04	0.07	0.491	0.54	0.73	091	91
Fuzzy C-means	0.09	0.05	0.5	0.536	0.74	0.94	495
Learning Vector Quantization	0.3	0.301	0.491	0.5	0.7375	0.679	176
Genetic Algorithm	0.04	0.006	0.49	0.53	0.738	0.941	59

Table 3.4: Partial Set of Predicted Risk Values for Nasa60 dataset without security attributes

	1	1	3	4	5	6	7	8	9	10
Kohonen	119.89	103.2764	91.806	108.0078	99.9625	117.41	93.53	137.004	117.0938	116.48
K-means	116.48	115.54	105.37	135.24	124.68	121.40	118.67	124.67	102.113	125.61
Random Centers	103.95	105.72	104.32	100.87	109.32	105.63	114.64	121.39	107.43	106.23
Fuzzy C-means	125.93	98.57	121.60	110.61	117.82	99.93	92.87	92.57	94.188	105.74
LVQ	110.87	100.58	102.38	110.66	95.162	117.3781	103.29	94.86	108.27	108.466
Genetic Algorithm	108.466	108.466	104.776	116.07	95.64	93.49	99.14	95.64	96.188	112.02

Table 3.4 and table 3.5 shows the partial set of predicted and actual risk values with security attribute.

3.8.2 Comparision between risk values with Security Attribute and risk values without Security Attribute

Table 3.6 shows the comparison results.

Table 3.5: Partial Set of Actual Risk Values for Nasa60 dataset with security attributes

	1	1	3	4	5	6	7	8	9	10
Kohonen	119.87	103.77	92.18	108.52	99.39	116.22	94.62	138.804	115.154	110.29
K-means	116.91	115.88	105.783	135.795	125.175	121.83	119.11	124.96	103.56	123.12
Random Centers	105.45	105.99	106.23	102.56	109.88	105.24	112.27	121.67	107.86	106.30
Fuzzy C-means	125.96	98.62	121.98	112.10	118.25	98.98	92.95	94.29	94.82	103.37
LVQ	108.13	103.8	102.18	119.52	92.21	116.15	100.29	92.63	105.7	102.21
Genetic Algorithm	105.1	104.3	103.92	110.52	92.54	90.9	98.64	95.04	95.5	110.66

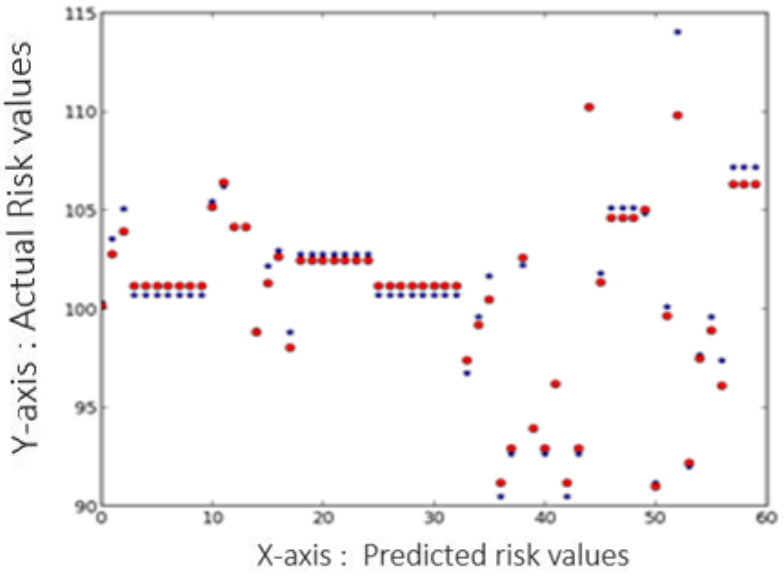

Figure 3.3: Partial plot of risk values with and without security in Kohonen neural network

It is clear from table 3.6 that there is a significant amount of decrease in the risk values on the addition of security attributes. And it is evident that the model K-means RBFN stands out in performance, followed by Kohonen neural network.

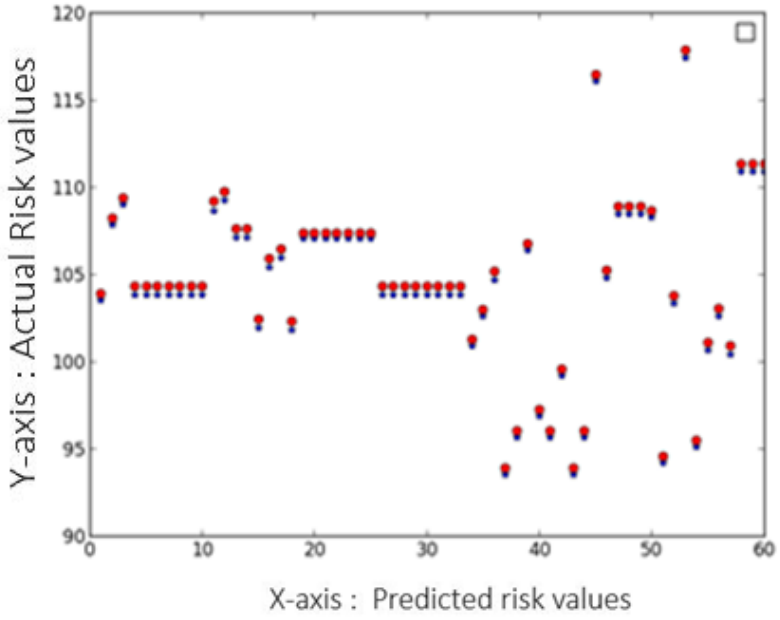

Figure 3.4: Partial plot of risk values with and without security in Radial Basis neural network

Table 3.6: Experimental Values with Security Attribute

	MMRE	MSE	PRED(0.25)	PRED(0.5)	PRED(0.75)	Correlation	Epochs
experimental values with Security Values							
Kohonen	**0.2**	**0.3**	**0.6**	**0.74**	**0.91**	**0.73**	**252**
K-means RBFN	**0.08**	**0.28**	**0.492**	**0.61**	**0.75**	**0.89**	**79**
Random Centers RBFN	**0.04**	**0.07**	**0.491**	**0.54**	**0.73**	**0.91**	**91**
Fuzzy C-means RBFN	**0.09**	**0.05**	**0.5**	**0.536**	**0.74**	**0.94**	**495**
Learning Vector Quantization	**0.3**	**0.301**	**0.491**	**0.5**	**0.7375**	**0.679**	**176**
Genetic Algorithm	**0.04**	**0.06**	**0.49**	**0.53**	**0.736**	**0.941**	**59**
Experimental values without security Values							
Kohonen	**0.12**	**0.26**	**0.72**	**0.79**	**0.96**	**0.65**	**226**
K-means RBFN	**0.06**	**0.19**	**0.53**	**0.66**	**0.81**	**0.82**	**132**
Random Centers RBFN	**0.06**	**0.11**	**0.491**	**0.54**	**0.73**	**0.91**	**91**
Fuzzy C-means RBFN	**0.07**	**0.036**	**0.61**	**0.68**	**0.772**	**0.88**	**372**
Learning Vector Quantization	**0.195**	**0.265**	**0.615**	**0.662**	**0.7375**	**0.679**	**105**
Genetic Algorithm	**0.05**	**0.012**	**0.72**	**0.85**	**0.88**	**0.963**	**59**

Chapter 4

Conclusion and Future Work

4.0.3 Conclusion

The contribution secured the issues identified with determination of viable risk analysis of projects with security factors for the software development. A methodology, which portrays an alternative of effort consideration, that structures an essential component for risk assessment identified with the created product security on asked for security level, was planned.

From the proposed work, it is demonstrated for low level of security, the adjustment in risk values was unimportant. In case of software products that required assurance levels between EAL-5 and EAL-7, the adjustment in risk values were considerable. Expert COCOMO gives more reasonable results because of more exact alignment of assessment of SECU attribute that influences upon the level of complexity.

A product development can be thought to be one of the more riskier projects in the current period. Driven by the instability of client prerequisites, the procedure (individuals, technique, devices), and the improbable way of the acquiring item, the software project development can run a high risk condition. In this paper, it was demonstrated that addition of security factors into a software project resulted in a decrease of the risk values. Among the neural network techniques implemented, it

is proved that Genetic Algorithm provided the most promising result.

4.0.4 Future Work

Future work may include improvement of the security levels. There are 7 Evaluation Assurance Levels. New Assurance Levels can be proposed for the assessment of risks. Moreover, machine learning techniques like decision trees, random forests, Bayes classifier etc can be implemented to evaluate the risk values of the fuzzy-Excom model.

Bibliography

[1] Jitka Kreslíková and Jana Sedláčková. Security factors in effort estimation. In *Information Systems Architecture and Technology - IT Models in Management Process*, page 11. Wroclaw University of Technology, 2010.

[2] Roy Schmidt, Kalle Lyytinen, and Paul Cule Mark Keil. Identifying software project risks: an international delphi study. *Journal of management information systems*, 17(4):5–36, 2001.

[3] Staffs Keele. Guidelines for performing systematic literature reviews in software engineering. Technical report, Technical report, EBSE Technical Report EBSE-2007-01, 2007.

[4] Mark Paulk. *Capability maturity model for software*. Wiley Online Library, 1993.

[5] Barry Boehm. *Software risk management*. Springer, 1989.

[6] Linda Wallace, Mark Keil, and Arun Rai. Understanding software project risk: a cluster analysis. *Information & Management*, 42(1):115–125, 2004.

[7] Shashank Mouli Satapathy, Mukesh Kumar, and Santanu Kumar Rath. Fuzzy-class point approach for software effort estimation using various adaptive regression methods. *CSI transactions on ICT*, 1(4):367–380, 2013.

[8] Lionel C Briand, Khaled El Emam, Dagmar Surmann, Isabella Wieczorek, and Katrina D Maxwell. An assessment and comparison of common software

cost estimation modeling techniques. In *Proceedings of the 21st international conference on Software engineering*, pages 313–322. ACM, 1999.

[9] Steve Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In *Real-Time Systems Symposium, 2007. RTSS 2007. 28th IEEE International*, pages 239–243. IEEE, 2007.

[10] Barry Boehm, Bradford Clark, Ellis Horowitz, Chris Westland, Ray Madachy, and Richard Selby. Cost models for future software life cycle processes: Cocomo 2.0. *Annals of software engineering*, 1(1):57–94, 1995.

[11] Ali Idri, Alain Abran, and Laila Kjiri. Cocomo cost model using fuzzy logic. In *7th International Conference on Fuzzy Theory & Techniques*, volume 27, 2000.

[12] Dan Port and Marcel Korte. Comparative studies of the model evaluation criterions mmre and pred in software cost estimation research. In *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, pages 51–60. ACM, 2008.

[13] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. *Noise reduction in speech processing*, volume 2. Springer, 2009.

[14] Barry W. Boehm. A spiral model of software development and enhancement. *Computer*, 21(5):61–72, 1988.

[15] Warren E Walker, Poul Harremoës, Jan Rotmans, Jeroen P van der Sluijs, Marjolein BA van Asselt, Peter Janssen, and Martin P Krayer von Krauss. Defining uncertainty: a conceptual basis for uncertainty management in model-based decision support. *Integrated assessment*, 4(1):5–17, 2003.

[16] Daya Gupta and Mohd Sadiq. Software risk assessment and estimation model. In *Computer Science and Information Technology, 2008. ICCSIT'08. International Conference on*, pages 963–967. IEEE, 2008.

[17] Mark Keil, Paul E Cule, Kalle Lyytinen, and Roy C Schmidt. A framework for identifying software project risks. *Communications of the ACM*, 41(11):76–83, 1998.

[18] Brian Patrick Green and Jae Hwa Choi. Assessing the risk of management fraud through neural network technology. *Auditing: A Journal of Practice & Theory*, 16(1):14, 1997.

[19] Michael R Lyu et al. *Handbook of software reliability engineering*, volume 222. IEEE computer society press CA, 1996.

[20] Zan Huang, Hsinchun Chen, Chia-Jung Hsu, Wun-Hwa Chen, and Soushan Wu. Credit rating analysis with support vector machines and neural networks: a market comparative study. *Decision support systems*, 37(4):543–558, 2004.

[21] Thomas G Calderon and John J Cheh. A roadmap for future neural networks research in auditing and risk assessment. *International Journal of Accounting Information Systems*, 3(4):203–236, 2002.

[22] Barry W Boehm, Ray Madachy, Bert Steece, et al. *Software Cost Estimation with Cocomo II with Cdrom*. Prentice Hall PTR, 2000.

[23] Mukesh Vijay Goyal, Shashank Mouli Satapathy, and Santanu Kumar Rath. An assessment and comparison of common software cost estimation modeling techniques. In *Proceedings of International Conference on Computing, Communication and Automation (ICCCA)*. IEEE, 2015.

[24] Stephen Haag, MmK Raja, and Lawrence L Schkade. Quality function deployment usage in software development. *Communications of the ACM*, 39(1):41–49, 1996.

[25] Christian Sven Collberg, Clark David Thomborson, and Douglas Wai Kok Low. Obfuscation techniques for enhancing software security, December 23 2003. US Patent 6,668,325.

[26] Guttorm Sindre and Andreas L Opdahl. Eliciting security requirements with misuse cases. *Requirements engineering*, 10(1):34–44, 2005.

[27] S Naganandhini. Kohonen neural network. In *Networking: Proceedings of the International Conference on Computer Applications: 24-27 December 2010, Pondicherry, India*. Research Publishing Services, 2010.

[28] Mohamad T Musavi, Wahid Ahmed, Khue Hiang Chan, Kathleen B Faris, and Donald M Hummels. On the training of radial basis function classifiers. *Neural networks*, 5(4):595–603, 1992.

[29] JK Sing, DK Basu, M Nasipuri, and M Kundu. Improved k-means algorithm in the design of rbf neural networks. In *TENCON 2003. Conference on Convergent Technologies for the Asia-Pacific Region*, volume 2, pages 841–845. IEEE, 2003.

[30] Dtreg predictive modelling software. `https://www.dtreg.com/`.

[31] Shashank Mouli Satapathy, Mukesh Kumar, and Santanu Kumar Rath. Class point approach for software effort estimation using soft computing techniques. In *Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on*, pages 178–183. IEEE, 2013.

[32] Guang-Bin Huang and Chee-Kheong Siew. Extreme learning machine with randomly assigned rbf kernels. *International Journal of Information Technology*, 11(1):16–24, 2005.

[33] Antonino Staiano, Roberto Tagliaferri, and Witold Pedrycz. Improving rbf networks performance in regression tasks by means of a supervised fuzzy clustering. *Neurocomputing*, 69(13):1570–1581, 2006.

[34] Usha Gupta and Manoj Kumar. Software effort estimation through clustering techniques of rbfn network. *IOSR Journal of Computer Engineering*, 14(3):58–62, 2003.

[35] Teuvo Kohonen. Learning vector quantization. In *Self-organizing maps*, pages 245–261. Springer, 2001.